Unstructured Mesh Generation Using Automatic Point Insertion and Local Refinement

PeiZong Lee
Institute of Information Science
Academia Sinica
Taipei, Taiwan, R.O.C.
leepe@iis.sinica.edu.tw

Chih-Hao Chang
Institute of Information Science
Academia Sinica
Taipei, Taiwan, R.O.C.
chchang@iis.sinica.edu.tw

Abstract

This paper is concerned with designing efficient algorithms for generating high-quality two-dimensional unstructured mesh. Because the most vital factor influencing the quality of mesh generation is point placement, we first propose a background quadtree to represent the density distribution within the computing domain that allows us to place well-spaced points. We then adopt the Bowyer-Watson Delaunay triangulation algorithm along with Steiner point insertion and local refinements. We identify constraints for placing new points that allow us to improve mesh quality incrementally. We also give procedures for performing local refinements which allow us to reduce obtuse angles and thus improve the mesh quality by reducing the aspect ratio and area ratio. A running example dealing with a blunt body in some flow domain is also presented to illustrate our method. Then we implement these algorithms to a three elements airfoil configuration.

Keywords: Adaptive mesh refinement, Delaunay triangulation, quadtree for density distribution, Steiner point insertion, unstructured mesh generation.

1 Introduction

Before numerical simulations are performed for such engineering problems as computational fluid dynamics and others, a structured or unstructured mesh is required to represent the computing domain. However, due to the increasingly complex geometry encountered, much attention has been devoted to the development of unstructured mesh [5]. Unstructured mesh allows greater flexibility in discretizing complex domains and enables straightforward implementation of adaptive mesh generation [14]. In practice, a high quality mesh should satisfy constraints made by the solver, such as criteria that measure the shape, size, or number of elements (triangles in our case). It is thus our goal in this paper to present a new method for generating high-quality two-dimensional unstructured mesh.

1.1 Motivation

When a finite element method or a finite volume method is adopted, the computing domain is tessellated into a mesh of triangular elements (triangles in our case) or other polygonal elements. We will use the term "triangles" instead of "elements" to avoid confusion in this paper. A solver is then used to approximate physical problems by means of discrete finite structures. The quality of triangles influences the convergence of the governing equations in the solver.

The quality measurement functions, which are frequently used, include the *aspect ratio*, *area ratio*, *edge ratio*, and the number of triangles. The aspect ratio of a triangle T is the ratio R_T/r_T, where R_T is the radius of the smallest circle containing T (circumcircle) and r_T is the radius of the largest circle contained in T (inscribed circle) [22]. The best aspect ratio is 2 due to the equilateral triangle.

The area ratio between two adjacent triangles is the ratio of the large area to the small area of these two triangles. The best area ratio is 1, where two adjacent triangles have the same area. The edge ratio of a triangle is the ratio of the length of the longest edge to the length of the shortest edge. The best edge ratio is 1, in the case of an equilateral triangle. The aspect ratio, area ratio, and edge ratio of a triangulation mesh are the largest (worst) aspect ratio, area ratio, and edge ratio among its triangles, respectively.

Because of the resolution requirement due to object geometry or interesting phenomena due to governing equations, *a part* but not all of the mesh in the computing domain is dense. However, to maintain a good area ratio, we also require a smooth change from dense triangles to sparse triangles. One of the most challenging problems is to generate the smallest number of triangles without losing any interesting phenomena in the simulation .

1.2 Summary of Results

We will accomplish the following goals.

1. We will propose a background quadtree to represent the density distribution of the computing domain which will guarantee that we will only need to generate a minimal number of triangles to achieve a high quality mesh.
2. Under our construction, using the Steiner point insertion strategy for Delaunay triangulation, we can prove that the aspect ratio is smaller than

4.31; the area ratio is smaller than 3; the edge ratio is smaller than 2; each angle in every triangle is greater than 30° and is less than 120°.

3. We will further propose methods for local refinements, which allow us to improve ill-conditioned triangles.
4. The number of triangles generated in the mesh has the same order as the number of leaves in the background quadtree for the density distribution.
5. The complexity of each step of our algorithm is linear with respect to the number of points inserted or the total number of triangles.

1.3 Related Works

Given a set of N points, existing Delaunay triangulation algorithms can construct a set of triangles that maximize the minimum angle based either on the divide and conquer techniques [9, 19] or on the convex-hull techniques [18] in time $O(N \log N)$. However, Delaunay triangulation is only a method for constructing triangles; it does not guarantee their quality. Instead, point placement can determine the quality of the triangulation. Unfortunately, optimal point placement is still an open problem. It is difficult to determine the point placement which will not cause obtuse triangles if one does not generate the triangulation first.

We now survey four popular mesh generation algorithms. The first quadtree method subdivides the computing domain orthogonally into four cells recursively, where each cell is a square, until the local geometry has been approximated. Then the triangles can be created by subdividing each cell in sequence using a simple and efficient way [25]. However, the mortal drawback is that points in the quadtree system are not amenable to the object geometry; therefore, extra effort is required to preserve the boundary curve by wrapping or adjusting triangles near the boundary.

The second Delaunay triangulation with Steiner point insertion method refers to the insertion of additional points into an existing triangulation in order to improve the quality of the triangulation [13]. If the circumcircle radius is greater than the density length in the center, it is natural to insert one point in the center [2, 4, 20, 23, 24]. Thus, based on this method, point placement matches the density distribution of the computing domain. Because new points inserted only affect the triangulation locally, it is particularly suitable for the adaptive solution strategy [14].

However, Delaunay triangulation also does not guarantee boundary integrity. Special treatments, such as edge swapping or adding more boundary points, are necessary if some boundary edges are not in the final triangulation. Other researchers also have proposed *constrained Delaunay triangulation* method to ensure that all boundary edges are in the final triangulation [6, 10, 21]. However, the quality of the mesh still cannot be guaranteed.

Unlike the two methods described above, the third method guarantees boundary integrity. In the advancing-front method, one first provides the initial fronts on the boundary curves. New points are inserted ahead of the selected front one at a time. New triangles are constructed by joining each new point with existing fronts. The current front is then removed from the list since it is now obscured by the new triangle, and new fronts are generated from the new triangles. This process continues until the entire domain have been tessellated [11, 16]. Placement of a new point is first determined by the prescribed point distribution function, which is defined in the background mesh, resulting in a triangle of optimal size and shape. In general, the advancing-front method can generate smooth high-quality unstructured mesh in most of the domain.

However, difficulties arise when two fronts or many fronts are encountered. In this case, the newly placed point may intersect with existing fronts, or this point may be too close to the existing points. Since there is no general strategy for generating the new triangle, one is forced to check all the near-by fronts and points to avoid congestion. The examination process may be very time-consuming. A hybrid approach is to adopt the advancing front method to place points in the computing domain, then the unstructured mesh is accomplished using an existing Delaunay triangulation algorithm [12, 17]. Alternatively, in this paper, we will use the advancing front method to generate several boundary fronts in order to preserve boundary integrity while using Delaunay triangulation with the Steiner point insertion method to generate other internal points.

The fourth sphere-packing based method is built upon the equivalence between a well-shaped mesh and a well-spaced point set [15, 22]. First, a balanced quadtree refinement is applied. Second, for each leaf cell, a set of over-sampling random points is generated. Third, for each point, a circle, whose center is the point and whose radius is equal to one half of the estimated local spacing, is generated. Because these circles might overlap, they are reduced to an undirected graph, where each circle is represented by a node. If two circles overlap, then there is an edge connecting the corresponding two nodes. The final set of points can then be generated by means of a maximal independent set of circles. After that, the Delaunay triangulation method is adopted to generate the mesh. It is shown that the worst case edge ratio obtained using this method is 2 or even worse depending on the sampling points [22].

On the theoretical side, Baker *et al.* gave an algorithm for triangulating the interior of a polygon with angles between 13° and 90° [1]. Chew further presented an algorithm which under uniform density distribution, could improve the results such that there were no angles less than 30° [7]. Bern *et al.* showed that the balanced quadtree refinement generated a well-shaped mesh. They proposed algorithms which produced no angles less than 18.4° [3]. Miller and Teng *et al.* also proved that the sphere-packing based method generated a well-shaped mesh [15, 22]. However, in practice, we have found that the results can be better if we further apply local refinements.

The rest of this paper is organized as follows. In Section 2, we define the density distribution of points (or triangles) within the computing domain. In Section 3, we present our Steiner point insertion method and some optimizations. We also give upper bounds of the aspect ratio, area ratio, and edge ratio. The complete proofs can be seen in Section 4. In Section 5, we present experimental studies dealing with a blunt body in some flow domain and a three elements airfoil. Finally, some concluding remarks are given in Section 6.

2 Representing Density Distribution by a Quadtree

Because the solution near extreme points or near the object boundary may vary greatly, these regions normally require much smaller triangles than do other regions. Additionally, in order to maintain a good area ratio, edge lengths between neighboring triangles should not change too much. We will now propose a quadtree to represent the density distribution.

Figure 1-(d) shows a sample blunt body in some flow domain. The domain is decomposed by means of a quadtree representation. There are four types of cells, where each cell is a square. A white cell indicates that points can be placed in a systematic way, such as by using the quadtree method, some other tiling method, or simply the Steiner point insertion method. A boundary cell indicates that points are placed using the advancing-front method. An adjacent boundary cell indicates that points are placed using the Steiner point insertion method so as to guarantee a smooth transition from boundary cells to white cells. A cell not in the computing domain will not be placed any internal points.

2.1 Density Rank Table

We adopt the following rules according to Table 1 to decide the cell size.

density rank	density length	cell rank	cell's side length	triangles in a cell
1	Δx	1	$4\Delta x$	32
2	$k\Delta x$	1	$4\Delta x$	20
3	$k^2\Delta x$	1 or 2	$4\Delta x$ or $8\Delta x$	13 or 51
4	$2\Delta x$	2	$8\Delta x$	32
5	$2k\Delta x$	2	$8\Delta x$	20
6	$2k^2\Delta x$	2 or 3	$8\Delta x$ or $16\Delta x$	13 or 51
7	$4\Delta x$	3	$16\Delta x$	32
8	$4k\Delta x$	3	$16\Delta x$	20
9	$4k^2\Delta x$	3 or 4	$16\Delta x$ or $32\Delta x$	13 or 51
10	$8\Delta x$	4	$32\Delta x$	32
11	$8k\Delta x$	4	$32\Delta x$	20
12	$8k^2\Delta x$	4 or 5	$32\Delta x$ or $64\Delta x$	13 or 51
13	$16\Delta x$	5	$64\Delta x$	32
14	$16k\Delta x$	5	$64\Delta x$	20
15	$16k^2\Delta x$	5 or 6	$64\Delta x$ or $128\Delta x$	13 or 51
16	$32\Delta x$	6	$128\Delta x$	32

Table 1: The relations among the density rank, the density edge length, and the cell size. The fifth column is the estimated number of triangles in the corresponding cell.

1. The minimum edge length within the computing domain is Δx corresponding to density rank 1.
2. The density edge length corresponding to density rank i is $k^{i-1}\Delta x$, where $k = 1.26$, $k^2 = 1.5876$, and $k^3 = 2$. The factor k represents the expected area ratio between two adjacent triangles, which are in neighboring cells, respectively.
3. The side length of one cell, whose rank is j, is $2^{j-1}4\Delta x$. For example, the side lengths of cell ranks 1, 2, 3, 4, 5, and 6 are $4\Delta x$, $2 \times 4\Delta x$, $2^2 \times 4\Delta x$, $2^3 \times 4\Delta x$, $2^4 \times 4\Delta x$, and $2^5 \times 4\Delta x$, respectively.
4. A cell of rank 1 only contains points with density ranks from 1 to 3; a cell of rank j only contains points with density ranks from $3(j-1)$ to $3j$, where $j > 1$.
5. According to the cell rank and the density rank of points, one can estimate the number of triangles in a cell as shown in the fifth column of Table 1. This approximate number is equal to the number of isosceles rectangular triangles (the type 4 triangles shown in Figure ??-(b)) placed in a cell.

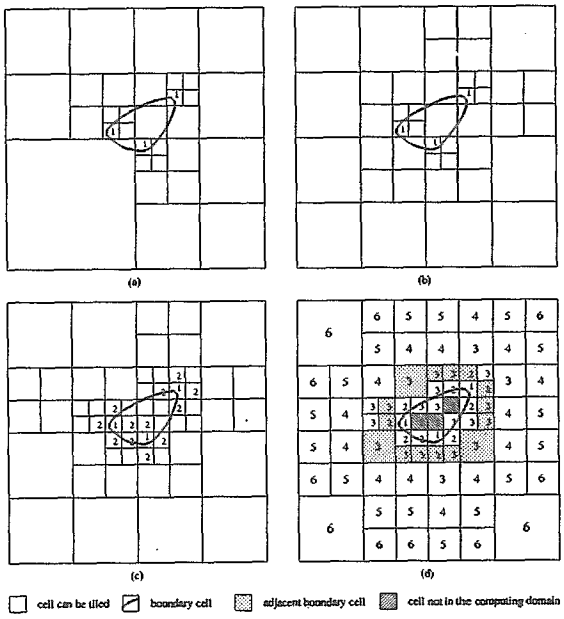

Figure 1: Illustration of the background quadtree for the density distribution: (a) The initial density constraints; (b) the balanced quadtree for the initial step; (c) the first intermediate step in assigning the density distribution; (d) the final quadtree for the density distribution.

2.2 Procedure for Assigning Density Distribution

From Table 1, we can assign density distribution within the computing domain. In Figure 1-(a), the initial density constraints are given. Three cells, each containing one extreme point, where the curvature is large, are assigned the highest density of density rank 1. Figure 1-(b) shows the balanced quadtree for the initial step. Figure 1-(c) illustrates the first intermediate step in assigning the density distribution. We use the advancing-front method starting from the cells A having the highest density (and, thus, the smallest density rank number); then, the density ranks of their

neighboring cells B, which were not visited before, are assigned to be

$$\min\{\text{original density rank of } B, (\text{density rank of } A) + 1\},$$

where the cell rank and the density rank have to satisfy constraints defined in Table 1. Otherwise, if the cell rank of B is greater than an expected rank, then cell B has to be further divided into four orthogonal small cells. On the other hand, if the cell rank of B is smaller than an expected rank, then the density rank of B is assigned the same value as the density rank of A. The final density distribution is shown in Figure 1-(d).

The background density quadtree implicitly maintains a well-spaced point placement. One immediate advantage is that we can use this background density quadtree to estimate the number of triangles in the final mesh. For example, if there are three rank-3 cells, all of whose density ranks are 6, then according to the fifth column in Table 1, about $3 \times 51 = 153$ triangles are placed in these three cells. Then, after accumulating the number of triangles in each cell, we obtain the estimated number of triangles in the final mesh, which is around 1694.

3 Delaunay Triangulation Using Steiner Point Insertion

Because Delaunay triangulation does not guarantee boundary integrity, we have to use a special treatment near boundary. First, boundary points are placed according to the density distribution as shown in Figure 4-(a). Second, initial triangulation for boundary points is performed; we then remove triangles which are out of the computing domain as shown in Figure 4-(b). Third, using the advancing-front method, we place two fronts of points near the boundary as shown in Figure 4-(c). The second front of points allows the algorithm to do local refinements but does not destroy the well-shaped quality of the first front of triangles. Therefore, the first front of triangles can guarantee the boundary integrity.

3.1 Algorithm Kernel

The algorithm kernel includes the following six steps.

Step 1: We perform basic Steiner point insertion based on the *circumcircle-center point placement strategy*. When the radius of the circumcircle of a triangle is greater than or equal to the density edge length (as defined in the second column in Table 1) w.r.t. the center, we insert one point in the center.

When the mesh quality needs to be improved, we iterate from Step 2 to Step 6.

Step 2: The first optimization tries to improve the well-spaced quality. Because three apexes of a triangle may lie in different cells of the background density quadtree, when the radius of the circumcircle is greater than or equal to the minimum density edge length w.r.t. three apexes, we insert one point in the center.

Step 3: The second optimization tries to improve the edge ratio. After performing Step 1 and Step 2, it may happen that even when all three apexes of a triangle lie on the same cell, the minimum edge length of the triangle is still less than the density edge length w.r.t. the center. In this case, the minimum edge length is at least greater than or equal to the density edge length of one of the adjacent cells. When the radius of the circumcircle is greater than or equal to the minimum edge length of that triangle, we insert one point in the center. Step 3 ensures that the edge ratio is less than 2, the minimum angle is greater than 30°, the maximum angle is less than 120°, and the aspect ratio is less than 4.31.

Step 4: The third optimization tries to improve the area ratio. It may happen that the radius R of the circumcircle of one adjacent triangle $\Delta p_1p_3p_4$ is greater than or equal to the minimum edge length e of the current triangle $\Delta p_1p_2p_3$; then, we insert one point p_0 in the center of the circumcircle of the adjacent triangle $\Delta p_1p_3p_4$ as shown in Figure 2-(a).

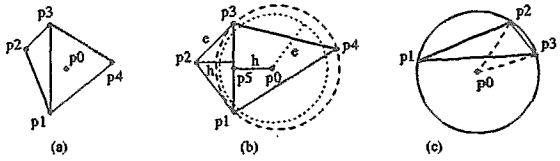

Figure 2: (a) When the circumcircle radius of $\Delta p_1p_3p_4$ is greater than or equal to the minimum edge length of its adjacent triangle $\Delta p_1p_2p_3$, a point p_0 can be inserted in the circumcircle center. (b) When $|\overline{p_4p_5}|$ is greater than or equal to $e + h$, where e is the minimum edge length and h is the height both of $\Delta p_1p_2p_3$, a point p_0 can be inserted. The dotted circle is centered at p_0 with a radius of e; the dashed circle is the circumcircle of $\Delta p_1p_3p_4$. (c) When point p_0 is the center of a circle, $\angle p_3p_0p_2 = 2\angle p_3p_1p_2$.

Step 5: The fourth optimization further tries to improve the area ratio. Suppose the height of the current triangle $\Delta p_1p_2p_3$ is h, and that the minimum edge length is e. Let point p_0 lie in the adjacent triangle $\Delta p_1p_3p_4$ and lie on the perpendicular bisection line (w.r.t. $\overline{p_1p_3}$) which passes through the central point $p_5 = (p_1 + p_3)/2$. In addition, the distance between p_0 and p_5 is h as shown in Figure 2-(b). Then, if the distance between p_5 and the opposite apex p_4 is greater than or equal to $e + h$, we insert one point p_0. Note that $|\overline{p_0p_1}|$, $|\overline{p_0p_3}|$, and $|\overline{p_0p_4}|$ are greater than or equal to e. Step 5 ensures that the area ratio is less than 3.

The advantage of using Steiner point insertion is that, amortizely, it only requires a constant amount of time to identify which triangle contains the inserted point. Thus, the time complexity of each Step i, for $1 \le i \le 6$, has the same order as the number of points inserted.

From Step 1 to Step 5, each constraint is finer than the previous one. This is because we want to improve the well-spaced quality gradually, we so try to minimize the number of inserted points. However, each optimization may conflict with other optimizations;

therefore, we have to iterate from Step 2 to Step 5 until the mesh quality is satisfactory. Even so, this algorithm eventually will terminate because the minimum edge length does not change throughout the execution, and the simulation domain is finite, which implies that we only can insert a finite number of points. Thus, the total time complexity of the algorithm kernel, which depends on the requirement of the mesh quality, is $O(rN)$, where N is the number of points and r is the number of iterations from Step 2 to Step 5.

3.2 Theorem Proving

In this subsection, we will give proofs for some theoretical results.

Theorem 1 *Suppose that a new point is inserted in the circumcircle center if the radius is greater than or equal to the minimum edge length of that triangle. Then, the edge ratio is less than 2, the minimum angle is greater than 30°, the maximum angle is less than 120°, and the aspect ratio is less than 4.31.*

Proof: Since the circumcircle contains the triangle, the length of the longest edge is less than or equal to the diameter, which is equal to twice the radius. Therefore, if the minimum edge length is less than or equal to one half of the longest edge length, then the radius is greater than or equal to the minimum edge length. Thus, we can insert one point in the center. We conclude that the longest edge length is less than twice the minimum (shortest) edge length. Hence, the edge ratio is less than 2.

Next, as shown in Figure 2-(c), in $\Delta p_1p_2p_3$, $\angle p_3p_1p_2$ is the smallest angle, and p_0 is the center of the circumcircle. Since $\angle p_3p_0p_2 = 2\angle p_3p_1p_2$, if $\angle p_3p_1p_2 \leq 30°$, then $\angle p_3p_0p_2 \leq 60°$. Therefore, the radius is greater than $|\overline{p_2p_3}|$. We thus can insert p_0 at the center. Hence, the smallest angle is greater than 30°, and this also implies that the largest angle is less than 120°.

Next, assume that the center of the inscribed circle of a triangle lies in the intersection of three angle bisection lines. An angle is small, which implies that the radius r_T of the inscribed circle is small too, but that the radius R_T of the circumcircle is large. The worst case aspect ratio R_T/r_T is about 4.31, which arises in the case where the three angles are 30°, 30°, and 120°, respectively. □

Theorem 2 *Suppose that the height of the current triangle $\Delta p_1p_2p_3$ is h, and that the minimum edge length is e. Let point p_0 lie in the adjacent triangle and lie on the perpendicular bisection line (w.r.t. $\overline{p_1p_3}$) which passes through the central point $p_5 = (p_1+p_3)/2$. In addition, the distance between p_0 and p_5 is h as shown in Figure 2-(b). If the distance between p_5 and the opposite apex p_4 is greater than or equal to $e+h$, then, it is feasible to insert point p_0; in addition, the area ratio is less than 3.*

Proof: First, we will show that it is feasible to insert p_0, such that the distance between p_0 and any other point is at least e. Because p_0 lies on the perpendicular bisection line (w.r.t. $\overline{p_1p_3}$) which passes through the central point $p_5 = (p_1 + p_3)/2$, and the distance between p_0 and p_5 is h, it follows that $|\overline{p_0p_1}| = |\overline{p_0p_3}|$. In addition, they are greater than or equal to the shorter length between $|\overline{p_1p_2}|$ and $|\overline{p_2p_3}|$; thus, they are greater than or equal to e. Next, from the triangulation inequality, $|\overline{p_0p_5}| + |\overline{p_0p_4}| \geq |\overline{p_5p_4}| \geq e + h$, we have $|\overline{p_0p_4}| \geq e$. Thus, the distance between p_0 and any point on the circumcircle of $\Delta p_1p_3p_4$ is at least e. For example, as shown in Figure 2-(b), the dotted circle, which represents the circle centered at p_0 with a radius of e, is within the dashed circle, which represents the circumcircle of $\Delta p_1p_3p_4$.

Next, we will show that the area ratio is less than 3. From Theorem 1, the minimum angle is greater than 30°; therefore, the height h is greater than one half of the minimum edge length e because $h > e\sin 30° = \frac{1}{2}e$. In addition, since $|\overline{p_4p_5}|$ is less than $e + h$ (otherwise, we can insert one point p_0 as mentioned in the last paragraph), the height from p_4 to the edge $\overline{p_1p_3}$ is less than $e + h$. Therefore, the area of $\Delta p_1p_3p_4$ is less than three times that of $\Delta p_1p_2p_3$. Hence, the area ratio is less than 3. □

Theorem 3 *The time complexity of each Step i in the algorithm kernel, for $1 \leq i \leq 6$, is linear with respect to the number of points.*

Proof: The time complexity of Delaunay triangulation algorithm includes the time for finding point locations for all inserted points and the time for creating triangles [8]. According to Lemma 9.11 in [8], the expected number of triangles created by Delaunay triangulation algorithm is at most $9N + 1$, where N is the number of points. Therefore, the time for creating triangles is linear w.r.t. the number of points. Thus, it is enough to show that the time for finding point locations is also linear w.r.t. the number of points.

We first show that the initial triangulation for boundary points can be done in a linear time. In two-dimensional space, the boundary of an object is either a (closed) curve or a polygon or a line. Because adjacent boundary points are placed according to the density distribution, the next boundary point inserted is either within the current triangle or within one of the current three neighboring triangles. Thus, except for the first boundary point which we use a sequential search, each of other point locations for adjacent boundary points can be found in a constant amount of time.

The proofs for all Step i, where $1 \leq i \leq 5$, are similar. Suppose that there has been some Delaunay triangulation. We check each triangle in sequence whether a point should be inserted at the circumcircle center (for Steps 1, 2, 3, and 4) or at other location within that triangle (for Step 5). According to the Bowyer-Watson Delaunay triangulation algorithm, we have to know which triangle contains the circumcircle center because the circumcircle center of a triangle maybe is not within the triangle itself. However, if the circumcircle center is not within its triangle, we can use the *right-hand rule* to decide a direction to which the

circumcircle center is beyond one of three edges.

Suppose that after t adjacent triangles, we finally find a triangle which contains that circumcircle center p_0. Then all corresponding circumcircles of these t triangles contain p_0. After applying the point insertion by the Bowyer-Watson Delaunay triangulation algorithm, all these t triangles are destroyed. Thus, the total time to identify all point locations is bounded by the number of triangles created. According to Lemma 9.11 in [8] again, it is $9N + 1$. This implies that Step i can be finished in a linear time w.r.t. the number of points.

Theorem 4 *The number of triangles generated in the mesh has the same order as the optimal one.*

Proof: The background quadtree, which is used to represent the density distribution of the computing domain, can be treated as a coarse-grain mesh. The quadtree method implicitly controls the area ratio such that the area ratio between two adjacent triangles is not greater than 2. If we require that for the optimal mesh, all triangles have no obtuse angles and the area ratio is bounded by 2, then each cell (leaf) of the quadtree should contain at least one point. Therefore, the number of triangles for the optimal mesh has at least the same order as the number of cells (leaves) in the quadtree.

However, as each cell can contain only a constant number of points (for a reference number, please refer to the fifth column of Table 1), the total number of points placed has the same order as the number of cells in the quadtree. From the Euler equation, the number of triangles is about twice the number of points [8]. Hence, the number of triangles generated in the mesh by our algorithm has the same order as the number of cells in the quadtree; thus, it also has the same order as the number of triangles for the optimal mesh. □

4 Local Refinements

Delaunay triangulation satisfies the empty circumcircle criterion, which states that no circumcircle of any triangle can contain a mesh point other than its forming points [8, 18, 19]. When the empty circumcircle criterion is not violated, we can apply local refinements to improve the mesh quality. However, in order to guarantee that the algorithm eventually will terminate, the minimum edge length among triangles can not be less than the minimum density length Δx as defined in Table 1.

When the aspect ratio or area ratio is bad or when there are obtuse angles, we will perform the following three local refinements.

Laplacian filter [24]: Suppose that point p_0 has edges connected to points $p_1, p_2, \ldots, p_n$. Then,

$$p_0^{\text{new}} = p_0^{\text{old}} + \omega(\Sigma_{i=1}^{n} p_i - p_0^{\text{old}})/n,$$

where ω is a factor and $0.05 \leq \omega \leq 0.25$. The Laplacian filter tries to maintain the well-spaced quality.

Reduce the obtuse angle: When a triangle has an obtuse angle $\angle p_1p_2p_3$ as shown in Figure 3-(a), we try to reduce the obtuse angle by 2θ, where $5° \leq \theta \leq 10°$, such that p_1 is moved to p_1' and p_3 is moved to p_3'. Note that, because $\angle p_1p_2p_3 - 2\theta > 60°$, $|\overline{p_1'p_3'}|$ is still greater than the minimum edge length. This optimization can improve the aspect ratio and area ratio when a triangle contains an obtuse angle. Of course, it also reduces obtuse angles.

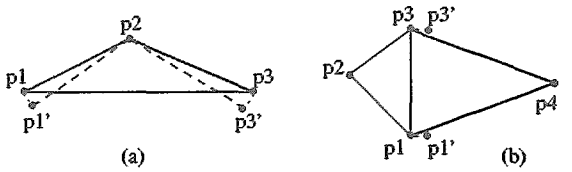

Figure 3: (a) The obtuse angle $\angle p_1p_2p_3$ is reduced to $\angle p_1'p_2p_3'$. (b) The shared edge $\overline{p_1p_3}$ is moved to $\overline{p_1'p_3'}$.

Move the shared edge: When the area ratio of two adjacent triangles is large, for example, when the area of $\Delta p_1p_2p_3$ is small and the area of $\Delta p_1p_3p_4$ is large as shown in Figure 3-(b), we move the shared edge $\overline{p_1p_3}$ toward the large-area triangle a distance of $h/10$ to $\overline{p_1'p_3'}$, where h is the height from p_4 to the shared edge $\overline{p_1p_3}$. Note that, under this refinement, all edge lengths are still greater than or equal to the original minimum edge length. This optimization can improve the area ratio.

5 Experimental Studies

We now continue the running example (as mentioned in Figures 1 and 4 in Sections 2 and 3, respectively,) that of dealing with the sample blunt body in some flow domain. The computing domain includes the planar coordinate which ranges within $[-32 : 32, -32 : 32]$. The minimum density edge length is 1. Initially, as shown in Figure 4-(a), points are placed on the boundary according to the density distribution. Figure 4-(b) shows the initial triangulation for the boundary points. Figure 4-(c) shows the triangulation after placing two fronts of points near the object boundary and the domain boundary.

Figure 4-(d) shows the triangulation for the first round after Step 1. The solid circle marks the triangle having the worst aspect ratio; the dashed circle marks the triangle having the worst area ratio; and the dash-dotted circle marks the triangle having the worst edge ratio. Then we recursively implement the optimizations from step 2 to step 5 in the kernel algorithms without the local refinement and show the result in Figure 4-(e). The aspect ratio, area ratio and the edge ratio of the resulting triangulation are 4.02, 2.84 and 1.96, which are all within the theoretical values of 4.31, 3.0 and 2.0. But there are still 148 obtuse triangles in the unstructured mesh.

If we include the local refinements during the mesh generation, the quality of the final mesh can be further improved, as shown in figure 4-(f). The aspect ratio of final triangulation is 3.78, the area ratio is 2.39, and the edge ratio is 1.94. In addition, the amount of obtuse triangles is reduced to 74.

The next example is a three elements airfoil for the

NASA energy efficient transport(EET). The computing domain is defined within $[-16:16, -6:16]$ and the minimum density edge length is 0.001. The mesh spacing is concentrated in the leading and trailing edge of each elements and varies substantially. Without local refinements, the generated mesh has the quality that aspect ratio is 4.23; area ratio is 2.93 and edge ratio is 1.99. In compariaion, the mesh with all the optimizations contains 33040 triangles and has the quality that aspect ratio is 4.19; area ratio is 2.88 and edge ratio is 1.94. The obtuse triangles are reduced from 3880 to 1806. The local refinements clearly provide improved quality. Figure 5-(a-c) show the finally generated unstructured mesh with local refinements.

6 Concluding Remarks

We have presented in this paper a new method to generate high-quality two-dimensional unstructured mesh. In order to guarantee well-spaced point placement, we require a smooth change of the density distribution, which can be represented by a quadtree. We can also prove that under our constraints, for Delaunay triangulation by means of Steiner point insertion, the aspect ratio is less than 4.31, the area ratio is less than 3, the edge ratio is less than 2, the minimum angle is greater than 30°, and the maximum angle is less than 120°. We have also proposed procedures for performing local refinements. These allow us to reduce obtuse triangles and improve the aspect ratio and the area ratio.

References

[1] B. S. Baker, E. Grosse and C. S. Rafferty, "Nonobtuse Triangulation of Polygons," Discrete and Comp. Geom., vol. 3, pp. 147–168, 1988.

[2] T. J. Baker, "Three Dimensional Mesh Generation by Triangulation of Arbitrary Points Sets," AIAA paper 87–1124, 1987.

[3] M. Bern, D. Eppstein and J. Gilbert, "Provably Good Mesh Generation," in Proc. 31st Annual Symp. on Foundations of Computer Science, pp. 231–241, 1990.

[4] A. Bowyer, "Computing Dirichlet Tessellations," The Computer Journal, vol. 24, no. 2, pp. 162–166, 1981.

[5] M. J. Chao, "Unstructured Grid Generation and Agglomeration Multigrid Method for the Euler Solutions over Complex Aircraft Configurations," Ph.D. thesis, National Cheng-Kung Univ., Taiwan, ROC, 1997.

[6] L. P. Chew, "Constrained Delaunay Triangulations," Algorithmica, vol. 4, pp. 97–108, 1989.

[7] L. P. Chew, "Guaranteed-quality Triangular Meshes," Technical Report TR-89-983, Cornell Univ., 1989.

[8] M. de Berg, M. van Kreveld, M. Overmars and O. Schwarzkopf, Computational Geometry: Algorithms and Applications, Springer, Berlin, 1997.

[9] D. T. Lee, "Two-Dimensional Voronoi Diagrams in the L_p-Metric," Journal of the Association for Computing Machinery, vol. 27, no. 4, pp. 604–618, October 1980.

[10] D. T. Lee and A. K. Lin, "Generalized Delaunay Triangulation for Planar Graphs," Discrete and Comp. Geom., vol. 1, pp. 201–217, 1986.

[11] R. Löhner and P. Parikh, "Generation of Three-Dimensional Unstructured Grids by The Advancing-Front Method," International Journal for Numerical Methods in Fluids, vol. 8, pp. 1135–1149, 1988.

[12] D. L. Marcum and N. P. Weatherill, "Unstructured Grid Generation Using Iterative Point Insertion and Local Reconnection," AIAA Journal, vol. 33, no. 9, pp. 1619–1625, 1995.

[13] D. J. Mavriplis, "Unstructured Mesh Generation and Adaptivity," ICASE Report 95-26, Inst. for Computer Applications in Science and Engineering, NASA Langley Research Center, Hampton, VA, April 1995.

[14] D. J. Mavriplis, "Unstructured Grid Techniques," Annu. Rev. Fluid. Mech., vol. 29, pp. 473–514, 1997.

[15] G. L. Miller, D. Talmor, S.-H. Teng and N. Walkington, "A Delaunay Based Numerical Method for Three Dimensions: Generation, Formulation, and Partition," in Proc. 27th Annu. ACM Symp. Theory Comput., pp. 683–692, 1995.

[16] P. Möller and P. Hansbo, "On Advancing Front Mesh Generation in Three Dimensions," International Journal for Numerical Methods in Engineering, vol. 38, pp. 3551–3569, 1995.

[17] J.-D. Müller, P. L. Roe and H. Deconinck, "A Frontal Approach for Internal Node Generation in Delaunay Triangulations," International Journal for Numerical Methods in Fluids, vol. 17, pp. 241–255, 1993.

[18] J. O'Rourke, Computational Geometry in C, Cambridge University Press, Cambridge, 1998.

[19] F. P. Preparata and M. I. Shamos, Computational Geometry, Springer, Berlin, 1985.

[20] S. Rebay, "Efficient Unstructured Mesh Generation by Means of Delaunay Triangulation and Bowyer-Watson Algorithm," Journal of Computational Physics, vol. 106, pp. 125–138, 1993.

[21] J. Ruppert, "A New and Simple Algorithm for Quality 2-Dimensional Mesh Generation," in Proc. 4th Annual ACM-SIAM Symposium on Discrete Algorithms, pp. 83–92, Austin, Texas, 1993.

[22] S.-H. Teng and C. W. Wong, "Unstructured Mesh Generation: Theory, Practice, and Perspectives," International Journal of Computational Geometry & Applications, 1999, to appear.

[23] D. F. Watson, "Computing the n-dimensional Delaunay Tessellation with Application to Voronoi Polytopes," The Computer Journal, vol. 24, no. 2, pp. 167–172, 1981.

[24] N. P. Weatherill, "Delaunay Triangulation in Computational Fluid Dynamics," Computers Math. Applic., vol. 24, no. 5/6, pp. 129–150, 1992.

[25] M. A. Yerry and M. S. Shephard, "Automatic Three-Dimensional Mesh Generation by the Modified Octree Technique," International Journal for Numerical Methods in Engineering, vol. 20, pp. 1965–1990, 1984.

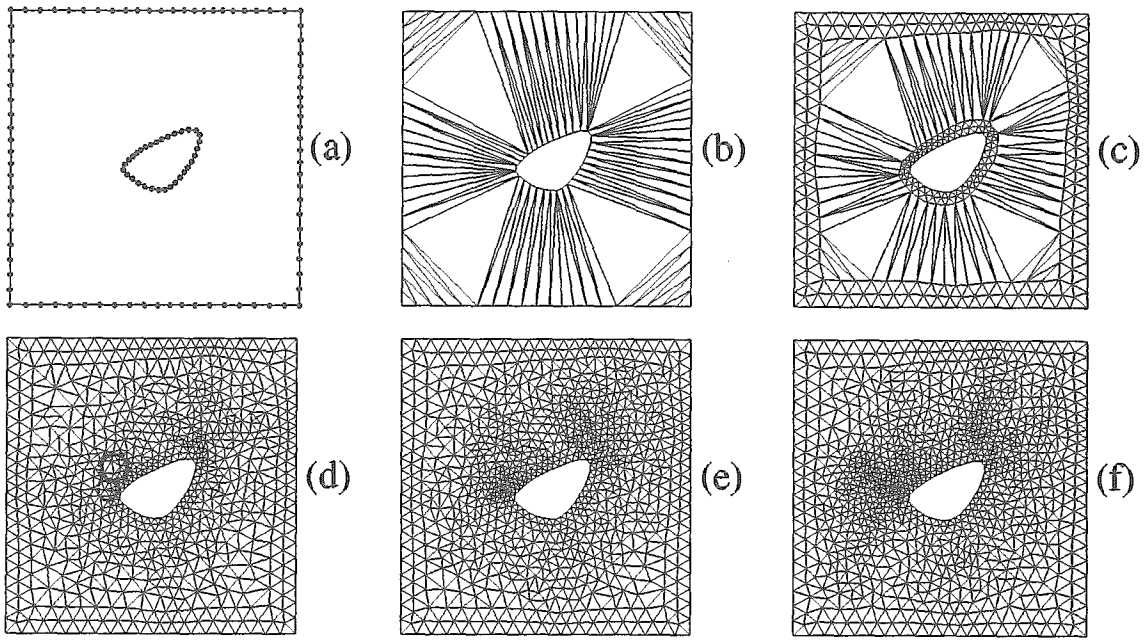

Figure 4: Delaunay triangulation by means of Steiner point insertion for the blunt body. (a) Initially, 125 points are placed along the object boundary and the domain boundary. (b) Initial triangulation for boundary points. (c) Triangulation after placing two fronts of points near the object boundary and the domain boundary. (d) Triangulation after Step 1 in the first round. (e) The final triangulation without local refinements. (f) The final triangulation with local refinements. The solid circle marks the triangle having the worst aspect ratio; the dashed circle marks the triangle having the worst area ratio; and the dash-dotted circle marks the triangle having the worst edge ratio.

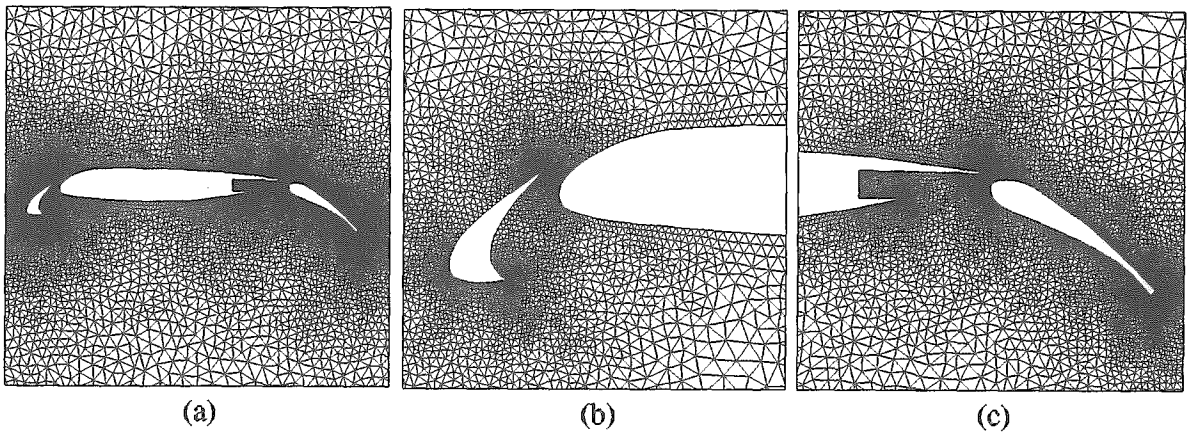

Figure 5: Delaunay triangulation by means of Steiner point insertion and local refinements for the three elements airfoil. (a) Mesh over view. (b) Mesh near front slat. (c) Mesh near rear flap.